K. Rampalli
Glyphzero Labs Inc.
Internet-Draft
Intended status: Standards Track
25 April 2026
Expires: 27 October 2026

PEDIGREE: Verifiable Delegation Identity for Agentic AI Systems
draft-rampalli-pedigree-00

Abstract

This document defines PEDIGREE (Per-Agent Delegation Identity with Governance-Enforced Execution), an identity and delegation framework for AI agents that extends the workload-identity model of SPIFFE (RFC 9542 / draft-ietf-wimse) with cryptographic per-hop delegation, monotonic scope attenuation enforced at mint and at verify, and dual-layer authority enforcement combining an operator-controlled ceiling with per-parent mandate narrowing.

PEDIGREE complements AAuth (draft-hardt-aauth-protocol) and AIP (draft-prakash-aip) by providing: (a) dual-enforcement semantics absent from both, (b) Cedar-policy mandates with static-analysis proofs of narrowing, (c) strict cryptographic parent-token re-verification that catches parent-swap attacks missed by append-only token chains, and (d) a native bridge to existing SPIFFE deployments.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 27 October 2026.

Copyright Notice

Table of Contents

Agentic AI systems spawn sub-agents dynamically. Current identity frameworks fail three tests:

1. *Attribution* -- when an orchestrator spawns a sub-agent to book flights and another to file expenses, both share the orchestrator's ambient credentials. Incident response cannot determine which sub-agent took which action.

2. *Attenuation* -- sub-agents typically inherit all parent authority. AAuth explicitly rejects scope attenuation (draft-hardt-aauth-protocol-01 Appendix C.3.7). Bearer-token and OAuth flows have no delegation narrowing primitive.

3. *Enforcement* -- even protocols that define attenuation (AIP) defer aggregate budget enforcement to "the orchestration platform at runtime" (draft-prakash-aip-00 Section 7.4), leaving a single-layer guard vulnerable to orchestrator compromise.

1.2. Non-Goals

* Replacing SPIFFE/WIMSE workload identity for in-cluster service-to-service auth. PEDIGREE is an upper layer that delegates from a SPIFFE-bearing root.

* Defining a user-authentication protocol. OIDC/OAuth flows bootstrap the root consent; PEDIGREE governs what happens after.

* Specifying an authorization policy language. PEDIGREE accepts Cedar (RECOMMENDED), Datalog, or Rego as mandate policy formats.

1.3. Related Work

The following table compares PEDIGREE with existing protocols across key delegation and enforcement dimensions.

Protocol	Sub-agent identity	Scope attenuation	Per-hop audit	Dual enforcement
AAuth (draft-hardt)	No	Explicitly rejected	Mission-level	No
AIP (draft-prakash)	Yes (aip:key:ed25519:)	Yes (Biscuit)	Mandatory context	No (budget punted)
SPIFFE SVID	Per-workload	None	Workload-level	No
ZeroID (open-source)	Yes (RFC 8693)	Yes	CAEP cascade	No
PEDIGREE (this document)	Yes (parent_chain)	Yes (Cedar, mint+verify)	Consent chain + SIEM	Yes (ceiling intersect mandate)

Table 1: Protocol Comparison

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

Root A human principal whose initial consent establishes the delegation chain.

Primary Agent A long-lived agent holding a signing keypair, directly delegated-to by the root.

Sub-Agent An ephemeral agent issued a PEDIGREE by a primary or another sub-agent.

PEDIGREE A signed JWT carrying principal identity, parent chain, mandate (policy set), public key, and lifetime.

SIT Suradar Identity Token -- the concrete wire format of a PEDIGREE, compatible with RFC 7519.

Mandate A typed policy set (Cedar by default) defining what actions an agent may perform on which resources.

Ceiling The deployment-level policy set defined by the operator; applies to all agents regardless of individual mandates.

Consent Chain An ordered sequence of principal-issued consents authorizing transitive delegation, with optional depth limits.

3. Identity Model

3.1. Identifier Schemes

PEDIGREE defines two identifier schemes:

Named (registry-backed) Format: pedigree:name:<deployment>/<agent-id>

Example: pedigree:name:glyphzero/trading-orchestrator-7f3a

Self-certifying (registry-free) Format: pedigree:key:ed25519:<multibase-public-key>

Example: pedigree:key:ed25519:z6MkpZ...

RECOMMENDED for sub-agents with lifetimes under 5 minutes.

3.2. Root of Trust

Each chain terminates at a root principal (human, identified via OIDC/OAuth). The primary agent’s PEDIGREE carries the root’s sub claim and a signature over the initial consent.

4. Token Format

4.1. Header

```
{
  "alg": "EdDSA",
  "typ": "pedigree+jwt"
}
```

4.2. Payload (SIT Claims)

The following is the set of claims carried in a SIT payload:

```
{
  "iss": "<issuer identity>",
  "sub": "<this agent's identity>",
  "jti": "<token uuid>",
  "iat": 1712345678,
  "exp": 1712349278,
  "pedigree_version": "0.1",
  "agent_pub": "<base64url ed25519 pubkey>",
  "parent_chain": ["<parent jti>", "<grandparent jti>"],
  "delegation_depth": 2,
  "scopes": ["read:cal", "write:cal"],
  "mandate": {
    "format": "cedar",
    "policy_set": "permit(principal, action == PedigreeAction::\"read_file\", resource);"
  },
  "consent_chain": ["<consent id>"],
  "ceiling_ref": "sha256:<operator ceiling hash>",
  "completion_bindable": true
}
```

4.3. Required Invariants

1. delegation_depth == len(parent_chain)

2. scopes MUST be a subset of the immediate parent's scopes.

3. exp - iat MUST NOT exceed the parent's remaining lifetime.

4. mandate.policy_set MUST narrow or equal the parent's mandate under the static-analysis check of Section 7.

5. Delegation

5.1. Mint

To mint a child SIT, the parent:

1. Verifies its own SIT is unexpired and not revoked.

2. Chooses a subset of its scopes.

3. Generates (or receives) the child's public key.

4. Computes a child mandate that is provably narrower than the parent's (see Section 7).

5. Appends its own jti to the child's parent_chain.

6. Signs the child JWT with its own private key.

5.2. Verify (Strict)

Verifiers MUST:

1. Cryptographically verify the leaf SIT against the issuer's public key.
2. Retrieve each parent JWT referenced by parent_chain and cryptographically verify each.
3. Check each parent's sub equals the corresponding parent_chain entry of the descendant.
4. Check each hop's scopes is a subset of the prior hop's scopes.
5. Check each hop's mandate is a narrowing of the prior hop's mandate via static analysis (Section 7).
6. Check the leaf's ceiling_ref hash matches the operator's current ceiling or a published prior version within grace period.

5.3. Parent-Swap Resistance

Because verification re-verifies each parent against the issuer's public key, an attacker cannot substitute a same-scope parent from a different chain. AIP's Biscuit append-only semantics trust block signatures and do not re-verify upstream; PEDIGREE's explicit re-verification catches this attack class.

6. Dual Enforcement

Every authorization decision requires both:

1. *Mandate:* the SIT's mandate.policy_set permits the action (parent's delegation narrowing).
2. *Ceiling:* the operator-deployed ceiling policy permits the action (applies to all agents).

Pseudocode:

```
allow = mandate.evaluate(action, resource)
     && ceiling.evaluate(action, resource)
```

The ceiling is identified by ceiling_ref (SHA-256 of the operator-signed ceiling policy document) and distributed out-of-band (or via a well-known URL).

This primitive is absent from AAuth, AIP, SPIFFE, and ZeroID.

7. Mandate Narrowing

7.1. Narrowing Check

A child mandate M_c narrows a parent mandate M_p if and only if:

```
forall (action, resource).
  M_c.permits(action, resource) => M_p.permits(action, resource)
```

7.2. Static Analysis

When mandate.format == "cedar", implementations MUST emit a static-analysis proof that M_c narrows M_p. Cedar’s published analyzer (AWS 2024) suffices. The proof artifact MAY accompany the SIT as a narrowing_proof field referencing the proof by content hash.

This static-proof primitive is absent from AIP (Biscuit semantics enforce narrowing imperatively but do not emit a verifiable artifact).

8. Completion Blocks

8.1. Purpose

A completion block cryptographically binds the outcome of an agent’s action to the delegation chain that authorized it, enabling downstream verifiers to prove a result came from an authorized chain without trusting out-of-band audit logs.

8.2. Format

```
{
  "jti": "<completion jti>",
  "parent_sit": "<sit jti>",
  "status": "completed",
  "result_hash": "sha256:<hex>",
  "verification_status": "self_reported",
  "cost_cents": 1234,
  "timestamp": "<rfc3339>",
  "signing_agent": "<agent identity>"
}
```

The status field MUST be one of: "completed", "failed", or "partial".

The verification_status field MUST be one of: "self_reported", "tool_verified", "peer_verified", or "human_verified".

The completion block is signed by the executing agent’s private key. Verifiers check the signature against agent_pub of the referenced SIT.

8.3. Chain Binding

A downstream agent receiving a result from an upstream agent SHOULD require the completion block alongside the result. The receiving agent MAY refuse to consume un-bound results.

9. Revocation

9.1. Cascade

Revoking a consent or a parent SIT invalidates every descendant in the chain. Implementations MUST walk the consent chain when evaluating revocation.

9.2. Shared Signals Framework

Implementations SHOULD emit OpenID SSF events (credential-change, session-revoked) on revocation to propagate to connected IdPs. See draft-rampalli-suradar-bindings for CAEP event schema.

10. Security Considerations

Parent-swap Mitigated by strict cryptographic re-verification (Section 5.3).

Scope escalation Mitigated by subset check at mint AND at verify.

Ceiling bypass Mitigated by dual enforcement (Section 6).

Audit evasion Mitigated by mandatory context on each delegation block and by cryptographic completion blocks (Section 8).

Key compromise Ephemeral self-cert keys (Section 3.1) limit blast radius to their TTL.

Replay jti uniqueness enforced by nonce store.

Transitive consent ConsentChain with TransitiveConsent=false blocks downstream delegation.

11. IANA Considerations

This document requests the following registrations:

11.1. JWT "typ" Header Value

Registration of the "pedigree+jwt" value in the "JSON Web Token Types" sub-registry of the "JSON Web Token (JWT)" registry:

Type Header Parameter Value pedigree+jwt

Specification Document This document, Section 4.1

11.2. URI Scheme

Provisional registration of the "pedigree" URI scheme with two defined forms:

* pedigree:name:<deployment>/<agent-id>

* pedigree:key:ed25519:<multibase-public-key>

11.3. Well-Known URI

Registration of the well-known URI /.well-known/pedigree-agent.json per RFC 8615.

12. References

12.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC7519] Jones, M., Bradley, J., and N. Sakimura, "JSON Web Token (JWT)", RFC 7519, DOI 10.17487/RFC7519, May 2015, <https://www.rfc-editor.org/info/rfc7519>.

[RFC8785] Rundgren, A., Jordan, B., and S. Erdtman, "JSON Canonicalization Scheme (JCS)", RFC 8785, DOI 10.17487/RFC8785, June 2020, <https://www.rfc-editor.org/info/rfc8785>.

[RFC9421] Backman, A., Richer, J., and M. Sporny, "HTTP Message Signatures", RFC 9421, DOI 10.17487/RFC9421, February 2024, <https://www.rfc-editor.org/info/rfc9421>.

[RFC9542] Gilman, E. and E. Baier, "SPIFFE Identity and Verifiable Identity Document", RFC 9542, DOI 10.17487/RFC9542, February 2024, <https://www.rfc-editor.org/info/rfc9542>.

12.2. Informative References

[I-D.hardt-aauth-protocol]
Hardt, D., "AAuth Protocol", Work in Progress, Internet-Draft, draft-hardt-aauth-protocol-01, March 2026, <https://datatracker.ietf.org/doc/html/draft-hardt-aauth-protocol-01>.

[I-D.prakash-aip]
Prakash, A., "Agent Interoperability Protocol", Work in Progress, Internet-Draft, draft-prakash-aip-00, March 2026, <https://datatracker.ietf.org/doc/html/draft-prakash-aip-00>.

[CEDAR] Amazon Web Services, "Cedar: A Language for Expressing Authorization Policies", https://www.cedarpolicy.com/, 2023.

[SPIFFE-SPEC]
CNCF SPIFFE Project, "SPIFFE/SPIRE Specification v1.0", https://spiffe.io/, 2023.

[RFC8615] Nottingham, M., "Well-Known Uniform Resource Identifiers (URIs)", RFC 8615, DOI 10.17487/RFC8615, May 2019, <https://www.rfc-editor.org/info/rfc8615>.

Appendix A. Comparison with AAuth and AIP

A detailed landscape comparison of PEDIGREE, AAuth, and AIP across all delegation, enforcement, and audit dimensions is maintained in the companion document: draft-rampalli-pedigree-landscape-comparison.

Key differentiators summarized:

* AAuth (draft-hardt-aauth-protocol-01) explicitly rejects scope attenuation and provides no sub-agent identity primitive. PEDIGREE adds both.

* AIP (draft-prakash-aip-00) provides sub-agent identity and Biscuit-based attenuation but defers budget enforcement to the orchestration platform and does not emit static-analysis proofs. PEDIGREE adds dual enforcement and Cedar-based narrowing proofs.

Appendix B. Reference Implementations

Go github.com/glyphzero/suradar -- production implementation covering L1-L3 of the SURADAR stack.

Python Planned.

TypeScript Planned.

Author's Address

Karthik Rampalli
Glyphzero Labs Inc.
Email: karthik@phantomcorgi.com